[ENGLISH]

Hello, Cloud Gurus, and welcome to this lesson,

where we'll cover DynamoDB scan and query API calls.

We'll begin with what is a query and what is a scan?

Have a quick demo.

We'll take a look

at performance improvements that we can make

and finish with exam tips.

So what is a query?

Well, a query operation finds items

in your table based on the primary key attribute

and also a distinct value to search for.

For example, if you're looking

for an item with a user ID of 212,

you're going to query the DynamoDB table

based on the primary key

of user ID and the value will be 212.

And that is going to return all the attributes for that

item. For example, first name, surname,

and email address, etc.

Now you can refine your query

by using an optional sort key name, and that's going

to refine the results if you have multiple results.

For example, if your sort key is a timestamp,

you can refine the query to only select items

with a timestamp of the last 7 days.

For instance, imagine you want

to a query the last time a user posted

in a forum--that kind of thing.

Now, by default, a query is going to return all

of the attributes that exist for the item that it returns.

However, you can use the ProjectionExpression

parameter if you want the query to return only

the specific attributes that you're looking for.

For example, if you only wanted to see the user ID

and email address

rather than all of the attributes

for that item in the table.

Now, query results are always sorted by the sort key,

and if the sort key is numeric, they are sorted

in numeric order, and by default that is in ascending order,

so 1, 2, 3, 4, etc.

If the sort key is an ASCII character, they are also sorted

by default in ascending order.

And if you need to, you can reverse the order

by setting the ScanIndexForward parameter to false.

And this ScanIndexForward parameter

only applies to queries.

It's just unfortunate that they've called its ScanIndex

Forward; it's not for scans, it only applies to queries.

By default, all queries are eventually consistent.

But you can set them to be strongly consistent,

and you will need to explicitly set the query

if you want it to be strongly consistent.

So moving on to scans, and a scan operation is completely

different because it examines every single item

in the table, and by default it's going

to return all of the items and all of the data attributes.

Once again, you can use the ProjectionExpression

parameter to refine your scan results so

that it only returns the attributes that you're looking for.

For instance, if you only want to see the email address

rather than all the attributes.

And you can also filter the results

of the scan once it's been run.

But really, the only way to understand the differences

between scans and queries

is if I actually show you in the console.

So here I am in the console.

The first thing I'm going to do is open

up the Cloud Shell, and you'll find it up here on the right.

I'll close that message down.

And we're going to run a couple of commands

to create a DynamoDB table that we can work with.

And if you remember, these are just the same commands we

used in a previous lesson to create a table.

And all of these commands are

in the resources for this lesson.

So first of all, we're just going to create a table,

and the table's going to be called ProductCatalog.

So copy the first command, come to the Cloud Shell,

we're going to paste it into the Cloud Shell,

verify the text, and select paste and hit Enter.

So that's going to create our table.

I'll clear my screen.

The second command is just going to download

the items.json file from GitHub.

And this is the file we'll use to populate our table.

So paste that and hit Enter, come back to our commands.

And then the third command is just going to populate the

table. And don't worry, you don't need to memorize the

syntax of these commands.

And if it's all worked, you should get this message

that just says UnprocessedItems.

And there should be no unprocessed items.

So now we can head to DynamoDB.

So search for DynamoDB in the search box,

select that,

select Tables on the left,

click on your ProductCatalog table.

And this is the table we just created.

And select Explore Items on the left.

And these are all the items in my ProductCatalog table.

So first of all, I'm going to run a query.

So select this dropdown, select Query,

and we're going to query the table

based on product ID, which is our partition key.

So I'm going to search for a product

with the product ID of 201,

so type 201 in the search box and hit Run.

And there we go.

It's found my product with the product ID of 201.

So that is a query.

What about a scan? Well, if we select Scan,

and then hit Run, it's going to give me the entire table.

But then if we scroll back up,

if I want to refine these results,

then I need to add a filter.

So let's filter based on one of the attributes

in our table, and we can filter based on price.

So the attribute name is going to be price,

the type is a number.

And let's search for all the items

in the table that have a price greater than 100.

So select Greater Than Or Equal To.

The value is 100 and select Run.

So that will apply a filter to my scan,

and now it's only displaying the items

with a price of greater or equal to 100.

But the one thing to remember

with a scan is that even when you are applying a filter,

it's first taking the entire table,

and then it's applying a filter on top

to show us only the results that we are looking for

based on our filter.

So what should we use, a query or a scan?

While a query is generally much more efficient than a scan

because a scan dumps the entire table and then it filters

out the unwanted items to provide the desired result.

And this adds the extra step

of removing the data that you don't want.

And as the table grows,

a scan operation is going to take longer and longer.

And you might be surprised to learn that a scan operation

on a large table can actually use up the provisioned

throughput using 1 single operation.

Now, there are some ways to improve your scan performance.

First of all, it's worth noting that a scan operation

processes data sequentially,

and it returns it in 1 megabyte increments

before moving on to retrieve the next 1 megabyte of data.

And it can only scan 1 partition at a time.

Now you can configure DynamoDB to use parallel scans instead

by logically dividing a table or index

into segments and then scanning each segment in parallel.

And this can be a good way to increase performance

if your database is not already really busy.

However, if your database is already really busy,

then it's best to avoid parallel scans

because if your table or index is

already incurring heavy read/write activity

from other applications, then parallel scans

can impact the other applications

that are trying to read from your table.

So what else can we do?

Another option is that you can isolate your scan operations

to only specific tables

and segregate them from your mission critical data

and your mission critical traffic,

even if that means writing data to 2 different tables.

Now for both queries and scans,

you can improve performance by setting a smaller page size.

For example, you could set the page size to return 40 items,

and that means that each operation is going to be smaller,

so it's going to run a larger number of smaller operations.

And the advantage of this

is that it avoids throttling the table,

and it should allow other requests

to succeed because you are not overloading the table

with 1 long operation.

However, in general, the rule of thumb is that

you should try to avoid scan operations if you possibly can.

Instead, design your tables

in such a way that you can use the Query, Get,

or BatchGetItem APIs instead.

So for the exam, just remember

that a scan operation is going to examine

every item in the table and return all the attributes.

For queries and scans,

you can use the ProjectionExpression parameter

to refine your results,

and you can also filter like we did earlier, to

show only the product with a price greater than 100.

A query operation finds items in the table

using only the primary key attribute.

So remember, when I searched the table

for a product ID of 201,

the query operation found that 1 single product for me.

Query results are always sorted

by the sort key if there is one,

and they're sorted by default in ascending order.

But you can reverse the order

by setting the ScanIndexForward parameter to false.

And this just reverses the order of your query results.

To improve performance of a query or scan,

set a smaller page size,

isolate scan operations to specific tables,

try parallel scans,

and remember that a query operation is generally

more efficient than a scan.

And if possible, you should design your tables

in such a way that you can use the Query, Get,

or BatchGetItem APIs and avoid using scans if you can.

So that is the end of this lesson.

Any questions, please let me know.

Otherwise, please join me in the next lesson.

Thank you.

[SPANISH]

Hola, Cloud Gurus, y bienvenidos a esta lección.

donde cubriremos el análisis de DynamoDB y las llamadas a la API de consulta.

Comenzaremos con ¿Qué es una consulta y qué es un escaneo?

Tener una demostración rápida.

echaremos un vistazo

en las mejoras de rendimiento que podemos hacer

y termine con consejos para el examen.

Entonces, ¿qué es una consulta?

Bueno, una operación de consulta encuentra elementos

en su tabla según el atributo de clave principal

y también un valor distinto para buscar.

Por ejemplo, si estás buscando

para un elemento con un ID de usuario de 212,

vas a consultar la tabla de DynamoDB

basado en la clave principal

de ID de usuario y el valor será 212.

Y eso va a devolver todos los atributos para eso.

artículo \_ Por ejemplo, nombre, apellido,

y dirección de correo electrónico, etc.

Ahora puedes refinar tu consulta

mediante el uso de un nombre de clave de ordenación opcional, y eso va

para refinar los resultados si tiene múltiples resultados.

Por ejemplo, si su clave de clasificación es una marca de tiempo,

puede refinar la consulta para seleccionar solo elementos

con una marca de tiempo de los últimos 7 días.

Por ejemplo, imagina que quieres

a una consulta la última vez que un usuario publicó

en un foro, ese tipo de cosas.

Ahora, de forma predeterminada, una consulta devolverá todos

de los atributos que existen para el elemento que devuelve.

Sin embargo, puede utilizar ProjectionExpression

parámetro si desea que la consulta solo devuelva

los atributos específicos que está buscando.

Por ejemplo, si solo desea ver el ID de usuario

y dirección de correo electrónico

en lugar de todos los atributos

para ese artículo en la tabla.

Ahora, los resultados de las consultas siempre se ordenan por clave de clasificación,

y si la clave de clasificación es numérica, se ordenan

en orden numérico, y por defecto es en orden ascendente,

entonces 1, 2, 3, 4, etc

Si la clave de ordenación es un carácter ASCII, también se ordenan

por defecto en orden ascendente.

Y si lo necesitas, puedes invertir el orden

configurando el parámetro ScanIndexForward en falso.

Y este parámetro ScanIndexForward

solo se aplica a las consultas.

Es desafortunado que lo hayan llamado ScanIndex

Adelante; no es para escaneos, solo se aplica a consultas.

De forma predeterminada, todas las consultas son finalmente coherentes.

Pero puede configurarlos para que sean fuertemente consistentes,

y deberá establecer explícitamente la consulta

si quieres que sea fuertemente consistente.

Entonces, pasando a los escaneos, una operación de escaneo es completamente

diferente porque examina cada elemento

en la tabla, y por defecto va

para devolver todos los elementos y todos los atributos de datos.

Una vez más, puede utilizar ProjectionExpression

parámetro para refinar los resultados de su escaneo para que

que solo devuelve los atributos que estás buscando.

Por ejemplo, si solo desea ver la dirección de correo electrónico

en lugar de todos los atributos.

Y también puedes filtrar los resultados.

del análisis una vez que se haya ejecutado.

Pero realmente, la única manera de entender las diferencias

entre escaneos y consultas

es si realmente te muestro en la consola.

Así que aquí estoy en la consola.

Lo primero que voy a hacer es abrir

suba el Cloud Shell, y lo encontrará aquí arriba a la derecha.

Cerraré ese mensaje.

Y vamos a ejecutar un par de comandos

para crear una tabla de DynamoDB con la que podamos trabajar.

Y si recuerdas, estos son los mismos comandos que

utilizado en una lección anterior para crear una tabla.

Y todos estos comandos son

en los recursos para esta lección.

En primer lugar, vamos a crear una tabla,

y la tabla se llamará ProductCatalog .

Así que copie el primer comando, vaya a Cloud Shell,

vamos a pegarlo en Cloud Shell,

verifique el texto, seleccione pegar y presione Entrar.

Eso va a crear nuestra tabla.

Borraré mi pantalla.

El segundo comando solo se va a descargar.

el archivo items.json de GitHub .

Y este es el archivo que usaremos para llenar nuestra tabla.

Así que pegue eso y presione Enter, regrese a nuestros comandos.

Y luego el tercer comando solo va a llenar el

mesa \_ Y no te preocupes, no necesitas memorizar el

sintaxis de estos comandos.

Y si todo funcionó, debería recibir este mensaje

eso solo dice UnprocessedItems .

Y no debe haber elementos sin procesar.

Así que ahora podemos dirigirnos a DynamoDB .

Así que busque DynamoDB en el cuadro de búsqueda,

seleccione eso,

seleccione Tablas a la izquierda,

haga clic en su tabla ProductCatalog .

Y esta es la tabla que acabamos de crear.

Y seleccione Explorar elementos a la izquierda.

Y estos son todos los artículos en mi tabla ProductCatalog .

Así que antes que nada, voy a ejecutar una consulta.

Seleccione este menú desplegable, seleccione Consulta,

y vamos a consultar la tabla

basado en la identificación del producto, que es nuestra clave de partición.

Así que voy a buscar un producto.

con el ID de producto de 201,

así que escriba 201 en el cuadro de búsqueda y presione Ejecutar.

Y ahí vamos.

Se encontró mi producto con el ID de producto de 201.

Entonces eso es una consulta.

¿Qué tal un escaneo? Pues si seleccionamos Escanear,

y luego presione Ejecutar, me dará la mesa completa.

Pero luego, si retrocedemos,

si quiero refinar estos resultados,

entonces necesito agregar un filtro.

Así que vamos a filtrar según uno de los atributos.

en nuestra tabla, y podemos filtrar según el precio.

Entonces, el nombre del atributo será el precio,

el tipo es un número.

Y vamos a buscar todos los elementos.

en la tabla que tienen un precio superior a 100.

Así que seleccione Mayor que o igual a.

El valor es 100 y seleccione Ejecutar.

Entonces eso aplicará un filtro a mi escaneo,

y ahora solo muestra los elementos

con un precio mayor o igual a 100.

Pero lo único que hay que recordar

con un escaneo es que incluso cuando está aplicando un filtro,

está tomando toda la mesa,

y luego está aplicando un filtro en la parte superior

para mostrarnos solo los resultados que buscamos

basado en nuestro filtro.

Entonces, ¿qué debemos usar, una consulta o un escaneo?

Mientras que una consulta es generalmente mucho más eficiente que un escaneo

porque un escaneo vuelca toda la tabla y luego filtra

el resultado deseado.

Y esto añade el paso extra

de eliminar los datos que no desea.

Y a medida que crece la mesa,

una operación de escaneo llevará más y más tiempo.

Y puede que se sorprenda al saber que una operación de escaneo

en una mesa grande puede agotar el aprovisionamiento

rendimiento utilizando 1 sola operación.

Ahora, hay algunas formas de mejorar el rendimiento de su escaneo.

En primer lugar, vale la pena señalar que una operación de escaneo

procesa los datos secuencialmente,

y lo devuelve en incrementos de 1 megabyte

antes de pasar a recuperar el siguiente megabyte de datos.

Y solo puede escanear 1 partición a la vez.

Ahora puede configurar DynamoDB para usar escaneos paralelos en su lugar

dividiendo lógicamente una tabla o índice

en segmentos y luego escanear cada segmento en paralelo.

Y esta puede ser una buena manera de aumentar el rendimiento.

si su base de datos no está ya muy ocupada.

Sin embargo, si su base de datos ya está muy ocupada,

entonces es mejor evitar escaneos paralelos

porque si su tabla o índice es

ya incurre en una gran actividad de lectura/escritura

desde otras aplicaciones, luego escaneos paralelos

puede impactar las otras aplicaciones

que están tratando de leer de su mesa.

Entonces, ¿qué más podemos hacer?

Otra opción es que puede aislar sus operaciones de escaneo

solo a tablas específicas

y sepárelos de sus datos de misión crítica

y su tráfico de misión crítica,

incluso si eso significa escribir datos en 2 tablas diferentes.

Ahora, tanto para consultas como para escaneos,

puede mejorar el rendimiento configurando un tamaño de página más pequeño.

Por ejemplo, puede configurar el tamaño de la página para que devuelva 40 elementos,

y eso significa que cada operación va a ser más pequeña,

por lo que va a ejecutar una mayor cantidad de operaciones más pequeñas.

Y la ventaja de esto

es que evita estrangular la mesa,

y debería permitir otras solicitudes

para tener éxito porque no estás sobrecargando la mesa

con 1 operación larga.

Sin embargo, en general, la regla general es que

debe intentar evitar las operaciones de escaneo si es posible.

En su lugar, diseñe sus tablas

de tal manera que se puede utilizar Query, Get,

o BatchGetItem API en su lugar.

Entonces, para el examen, solo recuerda

que una operación de escaneo va a examinar

cada elemento de la tabla y devolver todos los atributos.

Para consultas y escaneos,

puede usar el parámetro ProjectionExpression

para refinar sus resultados,

y también puede filtrar como lo hicimos antes, para

mostrar solo el producto con un precio superior a 100.

Una operación de consulta encuentra elementos en la tabla.

usando solo el atributo de clave principal.

Así que recuerda, cuando busqué en la mesa

para un ID de producto de 201,

la operación de consulta encontró ese 1 solo producto para mí.

Los resultados de las consultas siempre se ordenan

por la clave de clasificación si hay una,

y se ordenan de forma predeterminada en orden ascendente.

Pero puedes invertir el orden.

configurando el parámetro ScanIndexForward en falso.

Y esto simplemente invierte el orden de los resultados de su consulta.

Para mejorar el rendimiento de una consulta o exploración,

establecer un tamaño de página más pequeño,

aislar las operaciones de escaneo en tablas específicas,

prueba escaneos paralelos,

y recuerde que una operación de consulta es generalmente

más eficiente que un escaneo.

Y si es posible, debe diseñar sus tablas

de tal manera que se puede utilizar Query, Get,

o BatchGetItem API y evite usar escaneos si puede.

Así que ese es el final de esta lección.

Para cualquier duda, por favor hágamelo saber.

De lo contrario, acompáñeme en la próxima lección.

Gracias.